deep-learning-pbl

April 29, 2024

[ ]:
```
%reset -f
```

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Importing Required Libraries

[ ]:
```
import matplotlib.pyplot as plt
import pandas as pd
import os
import numpy as np
import tensorflow as tf

from sklearn.model_selection import train_test_split
from tensorflow import keras
import tqdm
import cv2

from tensorflow.keras.regularizers import l2
from tensorflow.keras import layers
import seaborn as sns
import gzip

from tensorflow.keras import layers
from tensorflow.keras.regularizers import l2
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.python.client import device_lib
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping,␣
↪ReduceLROnPlateau
```

Loading Training/Validation Dataset

[ ]:
```
train_path='/content/drive/MyDrive/seg_train/'
```

[ ]:
```
os.listdir(train_path)
```

[ ]:
```
['buildings', 'forest', 'glacier', 'mountain', 'sea', 'street']
```

[ ]:
```
classes=os.listdir(train_path)
```

[ ]:
```
length={}
for n1 in classes:
    l=len(os.listdir(train_path+'/'+n1))
    length[n1]=l
length
```

[ ]:
```
{'buildings': 2191,
 'forest': 2271,
 'glacier': 2404,
 'mountain': 2512,
 'sea': 2274,
 'street': 2382}
```

[ ]:
```
sumi=0
for e1 in length:
    sumi+=length[e1]
sumi
```

[ ]:
```
14034
```

[ ]:
```
x=[]
y=[]
num_samples_per_class = 1500
sampled_count_per_class = {labels: 0 for labels in classes}
for labels in classes:
    pth=os.path.join(train_path,labels)
    for img in tqdm.tqdm(os.listdir(pth)):
      if sampled_count_per_class[labels] >= num_samples_per_class:
            break
      else:
          pic=cv2.imread(os.path.join(pth,img))
          x.append(pic)
          y.append(classes.index(labels))
      sampled_count_per_class[labels]+=1
```

```
68%|       | 1500/2191 [01:07<00:31, 22.07it/s]
66%|       | 1500/2271 [00:55<00:28, 27.05it/s]
62%|       | 1500/2404 [01:00<00:36, 24.66it/s]
60%|       | 1500/2512 [01:20<00:54, 18.54it/s]
66%|       | 1500/2274 [00:54<00:28, 27.36it/s]
63%|       | 1500/2382 [00:59<00:35, 25.09it/s]
```

Pre-Processing of Dataset

Resizing to 128x128

```
[ ]: for i in range(len(x)):
         img = np.array(x[i])
         x[i] = cv2.resize(img,(128,128))
```

```
[ ]: x = np.array(x)
     y = np.array(y)
```

```
[ ]: np.save('trainX',x)
     np.save('trainY',y)
```

```
[ ]: x.shape
```

```
[ ]: (9000, 128, 128, 3)
```

```
[ ]: y.shape
```

```
[ ]: (9000,)
```

```
[ ]: import numpy as np
     x = np.load("/content/trainX.npy")
     y = np.load("/content/trainY.npy")
```

Splitting into training and validation sets

```
[ ]: X_train, X_valid, y_train, y_valid = train_test_split(x, y, test_size=0.2,␣
      ↪random_state=42)
```

```
[ ]: X_train= X_train / 255.0
```

```
[ ]: X_valid = X_valid / 255.0
```

Deep CNN Model Initialization

```
[ ]: esp = 20
     callback=[
           tf.keras.callbacks.ModelCheckpoint(filepath="/content/drive/MyDrive/CNN/
      ↪weights/best_model.h5",␣
      ↪monitor='val_loss',save_best_only=True,verbose=1,save_weights_only=False,mode='auto'),
           tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=esp,␣
      ↪restore_best_weights=True),
           tf.keras.callbacks.ReduceLROnPlateau(factor=0.5, patience=5,min_lr=1e-4)
         ]
     model = tf.keras.models.Sequential([
         tf.keras.layers.Conv2D(16, (3, 3), activation='relu', input_shape=(128,␣
      ↪128, 3)),
         tf.keras.layers.Conv2D(16, (3, 3), activation='relu'),
```

```
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(6, activation='softmax')
])
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',␣
 ↪metrics=['accuracy'])
```

[ ]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 16)      448

 conv2d_1 (Conv2D)           (None, 124, 124, 16)      2320

 max_pooling2d (MaxPooling2  (None, 62, 62, 16)        0
 D)

 conv2d_2 (Conv2D)           (None, 60, 60, 32)        4640

 conv2d_3 (Conv2D)           (None, 58, 58, 32)        9248

 dropout (Dropout)           (None, 58, 58, 32)        0

 max_pooling2d_1 (MaxPoolin  (None, 29, 29, 32)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 27, 27, 64)        18496

 conv2d_5 (Conv2D)           (None, 25, 25, 64)        36928

 dropout_1 (Dropout)         (None, 25, 25, 64)        0
```

```
 max_pooling2d_2 (MaxPoolin  (None, 12, 12, 64)        0
 g2D)

 global_average_pooling2d (  (None, 64)                0
 GlobalAveragePooling2D)

 dense (Dense)               (None, 32)                2080

 dense_1 (Dense)             (None, 32)                1056

 dense_2 (Dense)             (None, 6)                 198

=================================================================
Total params: 75414 (294.59 KB)
Trainable params: 75414 (294.59 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Model Training & Validation

[ ]:
```
history = model.fit(X_train,y_train, epochs=150,batch_size=32,␣
  ↪validation_data=(X_valid,y_valid), callbacks=callback)
```

```
Epoch 1/150
224/225 [============================>.] - ETA: 0s - loss: 1.3142 - accuracy:
0.4177
Epoch 1: val_accuracy improved from -inf to 0.49556, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5

/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103:
UserWarning: You are saving your model as an HDF5 file via `model.save()`. This
file format is considered legacy. We recommend using instead the native Keras
format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(

225/225 [==============================] - 19s 44ms/step - loss: 1.3128 -
accuracy: 0.4183 - val_loss: 1.3129 - val_accuracy: 0.4956 - lr: 0.0010
Epoch 2/150
223/225 [============================>.] - ETA: 0s - loss: 1.0797 - accuracy:
0.5265
Epoch 2: val_accuracy improved from 0.49556 to 0.51833, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 29ms/step - loss: 1.0782 -
accuracy: 0.5272 - val_loss: 1.2472 - val_accuracy: 0.5183 - lr: 0.0010
Epoch 3/150
223/225 [============================>.] - ETA: 0s - loss: 0.9877 - accuracy:
0.5600
Epoch 3: val_accuracy improved from 0.51833 to 0.60222, saving model to
```

```
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 28ms/step - loss: 0.9860 -
accuracy: 0.5604 - val_loss: 1.1541 - val_accuracy: 0.6022 - lr: 0.0010
Epoch 4/150
224/225 [============================>.] - ETA: 0s - loss: 0.9321 - accuracy:
0.6042
Epoch 4: val_accuracy improved from 0.60222 to 0.62944, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 26ms/step - loss: 0.9329 -
accuracy: 0.6043 - val_loss: 1.0623 - val_accuracy: 0.6294 - lr: 0.0010
Epoch 5/150
225/225 [==============================] - ETA: 0s - loss: 0.8353 - accuracy:
0.6685
Epoch 5: val_accuracy improved from 0.62944 to 0.68000, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 29ms/step - loss: 0.8353 -
accuracy: 0.6685 - val_loss: 1.0712 - val_accuracy: 0.6800 - lr: 0.0010
Epoch 6/150
224/225 [============================>.] - ETA: 0s - loss: 0.8092 - accuracy:
0.6701
Epoch 6: val_accuracy did not improve from 0.68000
225/225 [==============================] - 6s 28ms/step - loss: 0.8088 -
accuracy: 0.6703 - val_loss: 1.1256 - val_accuracy: 0.6506 - lr: 0.0010
Epoch 7/150
225/225 [==============================] - ETA: 0s - loss: 0.7498 - accuracy:
0.7089
Epoch 7: val_accuracy improved from 0.68000 to 0.73111, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 30ms/step - loss: 0.7498 -
accuracy: 0.7089 - val_loss: 0.9394 - val_accuracy: 0.7311 - lr: 0.0010
Epoch 8/150
224/225 [============================>.] - ETA: 0s - loss: 0.6850 - accuracy:
0.7447
Epoch 8: val_accuracy did not improve from 0.73111
225/225 [==============================] - 6s 28ms/step - loss: 0.6843 -
accuracy: 0.7453 - val_loss: 0.9464 - val_accuracy: 0.6956 - lr: 0.0010
Epoch 9/150
225/225 [==============================] - ETA: 0s - loss: 0.6652 - accuracy:
0.7528
Epoch 9: val_accuracy improved from 0.73111 to 0.76222, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 30ms/step - loss: 0.6652 -
accuracy: 0.7528 - val_loss: 0.8450 - val_accuracy: 0.7622 - lr: 0.0010
Epoch 10/150
225/225 [==============================] - ETA: 0s - loss: 0.6222 - accuracy:
0.7681
Epoch 10: val_accuracy did not improve from 0.76222
225/225 [==============================] - 6s 28ms/step - loss: 0.6222 -
```

```
accuracy: 0.7681 - val_loss: 0.9146 - val_accuracy: 0.7400 - lr: 0.0010
Epoch 11/150
224/225 [============================>.] - ETA: 0s - loss: 0.5989 - accuracy:
0.7825
Epoch 11: val_accuracy improved from 0.76222 to 0.78778, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 31ms/step - loss: 0.5989 -
accuracy: 0.7828 - val_loss: 0.8656 - val_accuracy: 0.7878 - lr: 0.0010
Epoch 12/150
225/225 [==============================] - ETA: 0s - loss: 0.5726 - accuracy:
0.7910
Epoch 12: val_accuracy did not improve from 0.78778
225/225 [==============================] - 6s 27ms/step - loss: 0.5726 -
accuracy: 0.7910 - val_loss: 0.8749 - val_accuracy: 0.7622 - lr: 0.0010
Epoch 13/150
224/225 [============================>.] - ETA: 0s - loss: 0.5515 - accuracy:
0.7953
Epoch 13: val_accuracy did not improve from 0.78778
225/225 [==============================] - 6s 28ms/step - loss: 0.5519 -
accuracy: 0.7954 - val_loss: 0.9314 - val_accuracy: 0.7444 - lr: 0.0010
Epoch 14/150
224/225 [============================>.] - ETA: 0s - loss: 0.5447 - accuracy:
0.8002
Epoch 14: val_accuracy improved from 0.78778 to 0.79833, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 30ms/step - loss: 0.5448 -
accuracy: 0.7999 - val_loss: 0.8013 - val_accuracy: 0.7983 - lr: 0.0010
Epoch 15/150
225/225 [==============================] - ETA: 0s - loss: 0.5164 - accuracy:
0.8149
Epoch 15: val_accuracy did not improve from 0.79833
225/225 [==============================] - 6s 28ms/step - loss: 0.5164 -
accuracy: 0.8149 - val_loss: 0.7725 - val_accuracy: 0.7933 - lr: 0.0010
Epoch 16/150
224/225 [============================>.] - ETA: 0s - loss: 0.5119 - accuracy:
0.8119
Epoch 16: val_accuracy did not improve from 0.79833
225/225 [==============================] - 7s 30ms/step - loss: 0.5117 -
accuracy: 0.8118 - val_loss: 0.7751 - val_accuracy: 0.7811 - lr: 0.0010
Epoch 17/150
225/225 [==============================] - ETA: 0s - loss: 0.4977 - accuracy:
0.8197
Epoch 17: val_accuracy improved from 0.79833 to 0.80333, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 31ms/step - loss: 0.4977 -
accuracy: 0.8197 - val_loss: 0.7727 - val_accuracy: 0.8033 - lr: 0.0010
Epoch 18/150
225/225 [==============================] - ETA: 0s - loss: 0.4859 - accuracy:
```

```
0.8242
Epoch 18: val_accuracy did not improve from 0.80333
225/225 [==============================] - 6s 29ms/step - loss: 0.4859 -
accuracy: 0.8242 - val_loss: 0.7510 - val_accuracy: 0.7989 - lr: 0.0010
Epoch 19/150
225/225 [==============================] - ETA: 0s - loss: 0.4693 - accuracy:
0.8299
Epoch 19: val_accuracy did not improve from 0.80333
225/225 [==============================] - 6s 27ms/step - loss: 0.4693 -
accuracy: 0.8299 - val_loss: 0.7490 - val_accuracy: 0.7794 - lr: 0.0010
Epoch 20/150
225/225 [==============================] - ETA: 0s - loss: 0.4768 - accuracy:
0.8279
Epoch 20: val_accuracy did not improve from 0.80333
225/225 [==============================] - 6s 28ms/step - loss: 0.4768 -
accuracy: 0.8279 - val_loss: 0.7558 - val_accuracy: 0.7878 - lr: 0.0010
Epoch 21/150
224/225 [============================>.] - ETA: 0s - loss: 0.4470 - accuracy:
0.8383
Epoch 21: val_accuracy improved from 0.80333 to 0.81278, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 29ms/step - loss: 0.4470 -
accuracy: 0.8383 - val_loss: 0.7314 - val_accuracy: 0.8128 - lr: 0.0010
Epoch 22/150
225/225 [==============================] - ETA: 0s - loss: 0.4419 - accuracy:
0.8376
Epoch 22: val_accuracy improved from 0.81278 to 0.81833, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 31ms/step - loss: 0.4419 -
accuracy: 0.8376 - val_loss: 0.6967 - val_accuracy: 0.8183 - lr: 0.0010
Epoch 23/150
224/225 [============================>.] - ETA: 0s - loss: 0.4380 - accuracy:
0.8387
Epoch 23: val_accuracy did not improve from 0.81833
225/225 [==============================] - 6s 28ms/step - loss: 0.4381 -
accuracy: 0.8386 - val_loss: 0.7200 - val_accuracy: 0.7917 - lr: 0.0010
Epoch 24/150
225/225 [==============================] - ETA: 0s - loss: 0.4184 - accuracy:
0.8489
Epoch 24: val_accuracy did not improve from 0.81833
225/225 [==============================] - 6s 28ms/step - loss: 0.4184 -
accuracy: 0.8489 - val_loss: 0.6830 - val_accuracy: 0.8133 - lr: 0.0010
Epoch 25/150
225/225 [==============================] - ETA: 0s - loss: 0.4223 - accuracy:
0.8450
Epoch 25: val_accuracy improved from 0.81833 to 0.83278, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 29ms/step - loss: 0.4223 -
```

```
accuracy: 0.8450 - val_loss: 0.6331 - val_accuracy: 0.8328 - lr: 0.0010
Epoch 26/150
225/225 [==============================] - ETA: 0s - loss: 0.4078 - accuracy:
0.8501
Epoch 26: val_accuracy did not improve from 0.83278
225/225 [==============================] - 7s 29ms/step - loss: 0.4078 -
accuracy: 0.8501 - val_loss: 0.7041 - val_accuracy: 0.7767 - lr: 0.0010
Epoch 27/150
225/225 [==============================] - ETA: 0s - loss: 0.3947 - accuracy:
0.8553
Epoch 27: val_accuracy did not improve from 0.83278
225/225 [==============================] - 6s 27ms/step - loss: 0.3947 -
accuracy: 0.8553 - val_loss: 0.6905 - val_accuracy: 0.7944 - lr: 0.0010
Epoch 28/150
225/225 [==============================] - ETA: 0s - loss: 0.3878 - accuracy:
0.8610
Epoch 28: val_accuracy did not improve from 0.83278
225/225 [==============================] - 7s 29ms/step - loss: 0.3878 -
accuracy: 0.8610 - val_loss: 0.6372 - val_accuracy: 0.8089 - lr: 0.0010
Epoch 29/150
224/225 [============================>.] - ETA: 0s - loss: 0.3983 - accuracy:
0.8569
Epoch 29: val_accuracy did not improve from 0.83278
225/225 [==============================] - 6s 28ms/step - loss: 0.3983 -
accuracy: 0.8568 - val_loss: 0.7077 - val_accuracy: 0.7889 - lr: 0.0010
Epoch 30/150
225/225 [==============================] - ETA: 0s - loss: 0.3789 - accuracy:
0.8604
Epoch 30: val_accuracy did not improve from 0.83278
225/225 [==============================] - 6s 28ms/step - loss: 0.3789 -
accuracy: 0.8604 - val_loss: 0.6517 - val_accuracy: 0.8228 - lr: 0.0010
Epoch 31/150
224/225 [============================>.] - ETA: 0s - loss: 0.3393 - accuracy:
0.8807
Epoch 31: val_accuracy did not improve from 0.83278
225/225 [==============================] - 6s 27ms/step - loss: 0.3390 -
accuracy: 0.8807 - val_loss: 0.6103 - val_accuracy: 0.8328 - lr: 5.0000e-04
Epoch 32/150
225/225 [==============================] - ETA: 0s - loss: 0.3175 - accuracy:
0.8860
Epoch 32: val_accuracy did not improve from 0.83278
225/225 [==============================] - 7s 29ms/step - loss: 0.3175 -
accuracy: 0.8860 - val_loss: 0.6054 - val_accuracy: 0.8189 - lr: 5.0000e-04
Epoch 33/150
225/225 [==============================] - ETA: 0s - loss: 0.3104 - accuracy:
0.8883
Epoch 33: val_accuracy improved from 0.83278 to 0.83444, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
```

```
225/225 [==============================] - 7s 29ms/step - loss: 0.3104 -
accuracy: 0.8883 - val_loss: 0.5845 - val_accuracy: 0.8344 - lr: 5.0000e-04
Epoch 34/150
225/225 [==============================] - ETA: 0s - loss: 0.3030 - accuracy:
0.8901
Epoch 34: val_accuracy improved from 0.83444 to 0.85389, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 7s 31ms/step - loss: 0.3030 -
accuracy: 0.8901 - val_loss: 0.5417 - val_accuracy: 0.8539 - lr: 5.0000e-04
Epoch 35/150
225/225 [==============================] - ETA: 0s - loss: 0.2989 - accuracy:
0.8947
Epoch 35: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 27ms/step - loss: 0.2989 -
accuracy: 0.8947 - val_loss: 0.5965 - val_accuracy: 0.8283 - lr: 5.0000e-04
Epoch 36/150
224/225 [============================>.] - ETA: 0s - loss: 0.2940 - accuracy:
0.8947
Epoch 36: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 29ms/step - loss: 0.2953 -
accuracy: 0.8942 - val_loss: 0.5647 - val_accuracy: 0.8344 - lr: 5.0000e-04
Epoch 37/150
224/225 [============================>.] - ETA: 0s - loss: 0.2934 - accuracy:
0.8962
Epoch 37: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 27ms/step - loss: 0.2928 -
accuracy: 0.8964 - val_loss: 0.5337 - val_accuracy: 0.8478 - lr: 5.0000e-04
Epoch 38/150
225/225 [==============================] - ETA: 0s - loss: 0.2889 - accuracy:
0.8936
Epoch 38: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 27ms/step - loss: 0.2889 -
accuracy: 0.8936 - val_loss: 0.5336 - val_accuracy: 0.8406 - lr: 5.0000e-04
Epoch 39/150
223/225 [============================>.] - ETA: 0s - loss: 0.2772 - accuracy:
0.8976
Epoch 39: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 28ms/step - loss: 0.2755 -
accuracy: 0.8983 - val_loss: 0.5700 - val_accuracy: 0.8356 - lr: 5.0000e-04
Epoch 40/150
225/225 [==============================] - ETA: 0s - loss: 0.2769 - accuracy:
0.8975
Epoch 40: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 28ms/step - loss: 0.2769 -
accuracy: 0.8975 - val_loss: 0.5150 - val_accuracy: 0.8417 - lr: 5.0000e-04
Epoch 41/150
225/225 [==============================] - ETA: 0s - loss: 0.2761 - accuracy:
0.8983
```

```
Epoch 41: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 28ms/step - loss: 0.2761 -
accuracy: 0.8983 - val_loss: 0.5429 - val_accuracy: 0.8350 - lr: 5.0000e-04
Epoch 42/150
225/225 [==============================] - ETA: 0s - loss: 0.2728 - accuracy:
0.8982
Epoch 42: val_accuracy did not improve from 0.85389
225/225 [==============================] - 6s 28ms/step - loss: 0.2728 -
accuracy: 0.8982 - val_loss: 0.5097 - val_accuracy: 0.8428 - lr: 5.0000e-04
Epoch 43/150
225/225 [==============================] - ETA: 0s - loss: 0.2717 - accuracy:
0.9033
Epoch 43: val_accuracy did not improve from 0.85389
225/225 [==============================] - 7s 29ms/step - loss: 0.2717 -
accuracy: 0.9033 - val_loss: 0.5623 - val_accuracy: 0.8206 - lr: 5.0000e-04
Epoch 44/150
225/225 [==============================] - ETA: 0s - loss: 0.2647 - accuracy:
0.9032
Epoch 44: val_accuracy improved from 0.85389 to 0.86167, saving model to
/content/drive/MyDrive/CNN/weights/best_model.h5
225/225 [==============================] - 6s 29ms/step - loss: 0.2647 -
accuracy: 0.9032 - val_loss: 0.4927 - val_accuracy: 0.8617 - lr: 5.0000e-04
Epoch 45/150
225/225 [==============================] - ETA: 0s - loss: 0.2668 - accuracy:
0.9019
Epoch 45: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.2668 -
accuracy: 0.9019 - val_loss: 0.5379 - val_accuracy: 0.8428 - lr: 5.0000e-04
Epoch 46/150
225/225 [==============================] - ETA: 0s - loss: 0.2515 - accuracy:
0.9067
Epoch 46: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.2515 -
accuracy: 0.9067 - val_loss: 0.5359 - val_accuracy: 0.8344 - lr: 5.0000e-04
Epoch 47/150
225/225 [==============================] - ETA: 0s - loss: 0.2590 - accuracy:
0.9033
Epoch 47: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 29ms/step - loss: 0.2590 -
accuracy: 0.9033 - val_loss: 0.5146 - val_accuracy: 0.8489 - lr: 5.0000e-04
Epoch 48/150
224/225 [============================>.] - ETA: 0s - loss: 0.2430 - accuracy:
0.9121
Epoch 48: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.2430 -
accuracy: 0.9119 - val_loss: 0.5230 - val_accuracy: 0.8406 - lr: 5.0000e-04
Epoch 49/150
224/225 [============================>.] - ETA: 0s - loss: 0.2516 - accuracy:
```

```
0.9043
Epoch 49: val_accuracy did not improve from 0.86167
225/225 [==============================] - 7s 29ms/step - loss: 0.2517 -
accuracy: 0.9043 - val_loss: 0.5273 - val_accuracy: 0.8389 - lr: 5.0000e-04
Epoch 50/150
225/225 [==============================] - ETA: 0s - loss: 0.2118 - accuracy:
0.9206
Epoch 50: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.2118 -
accuracy: 0.9206 - val_loss: 0.4910 - val_accuracy: 0.8478 - lr: 2.5000e-04
Epoch 51/150
225/225 [==============================] - ETA: 0s - loss: 0.2029 - accuracy:
0.9274
Epoch 51: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.2029 -
accuracy: 0.9274 - val_loss: 0.5152 - val_accuracy: 0.8433 - lr: 2.5000e-04
Epoch 52/150
224/225 [============================>.] - ETA: 0s - loss: 0.2055 - accuracy:
0.9258
Epoch 52: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.2058 -
accuracy: 0.9257 - val_loss: 0.4602 - val_accuracy: 0.8500 - lr: 2.5000e-04
Epoch 53/150
224/225 [============================>.] - ETA: 0s - loss: 0.1993 - accuracy:
0.9254
Epoch 53: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.2004 -
accuracy: 0.9249 - val_loss: 0.4727 - val_accuracy: 0.8433 - lr: 2.5000e-04
Epoch 54/150
224/225 [============================>.] - ETA: 0s - loss: 0.1989 - accuracy:
0.9290
Epoch 54: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1998 -
accuracy: 0.9287 - val_loss: 0.5063 - val_accuracy: 0.8444 - lr: 2.5000e-04
Epoch 55/150
223/225 [============================>.] - ETA: 0s - loss: 0.1959 - accuracy:
0.9312
Epoch 55: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.1950 -
accuracy: 0.9318 - val_loss: 0.4690 - val_accuracy: 0.8494 - lr: 2.5000e-04
Epoch 56/150
224/225 [============================>.] - ETA: 0s - loss: 0.1960 - accuracy:
0.9240
Epoch 56: val_accuracy did not improve from 0.86167
225/225 [==============================] - 7s 29ms/step - loss: 0.1954 -
accuracy: 0.9243 - val_loss: 0.4648 - val_accuracy: 0.8478 - lr: 2.5000e-04
Epoch 57/150
224/225 [============================>.] - ETA: 0s - loss: 0.1911 - accuracy:
```

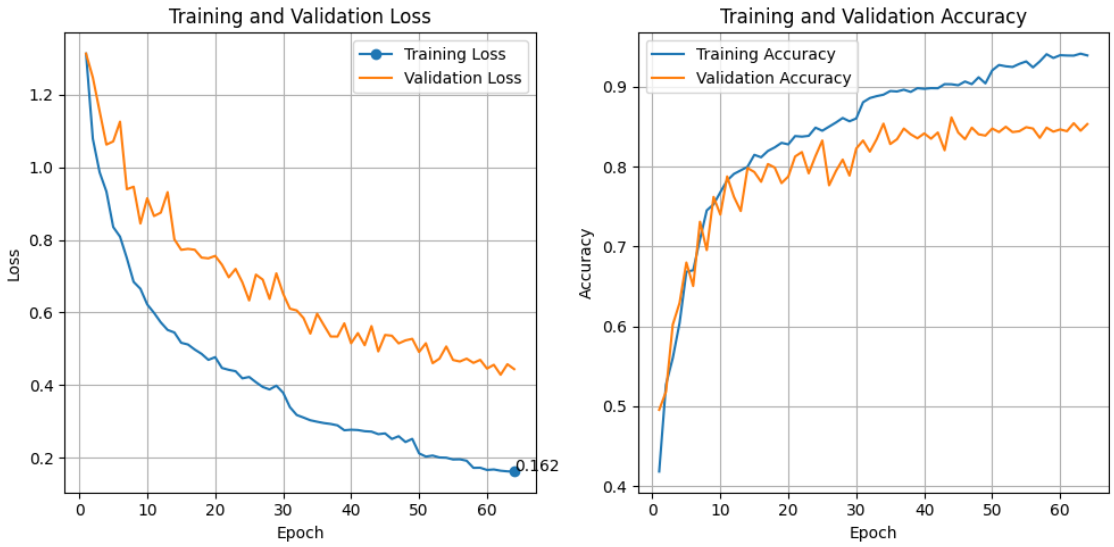
```
0.9318
Epoch 57: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.1915 -
accuracy: 0.9318 - val_loss: 0.4727 - val_accuracy: 0.8361 - lr: 2.5000e-04
Epoch 58/150
224/225 [============================>.] - ETA: 0s - loss: 0.1717 - accuracy:
0.9408
Epoch 58: val_accuracy did not improve from 0.86167
225/225 [==============================] - 7s 30ms/step - loss: 0.1718 -
accuracy: 0.9408 - val_loss: 0.4610 - val_accuracy: 0.8489 - lr: 1.2500e-04
Epoch 59/150
225/225 [==============================] - ETA: 0s - loss: 0.1722 - accuracy:
0.9361
Epoch 59: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1722 -
accuracy: 0.9361 - val_loss: 0.4694 - val_accuracy: 0.8439 - lr: 1.2500e-04
Epoch 60/150
225/225 [==============================] - ETA: 0s - loss: 0.1661 - accuracy:
0.9396
Epoch 60: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1661 -
accuracy: 0.9396 - val_loss: 0.4453 - val_accuracy: 0.8467 - lr: 1.2500e-04
Epoch 61/150
225/225 [==============================] - ETA: 0s - loss: 0.1674 - accuracy:
0.9392
Epoch 61: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 27ms/step - loss: 0.1674 -
accuracy: 0.9392 - val_loss: 0.4558 - val_accuracy: 0.8444 - lr: 1.2500e-04
Epoch 62/150
225/225 [==============================] - ETA: 0s - loss: 0.1640 - accuracy:
0.9390
Epoch 62: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1640 -
accuracy: 0.9390 - val_loss: 0.4284 - val_accuracy: 0.8544 - lr: 1.2500e-04
Epoch 63/150
223/225 [============================>.] - ETA: 0s - loss: 0.1628 - accuracy:
0.9413
Epoch 63: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1622 -
accuracy: 0.9415 - val_loss: 0.4574 - val_accuracy: 0.8450 - lr: 1.2500e-04
Epoch 64/150
224/225 [============================>.] - ETA: 0s - loss: 0.1616 - accuracy:
0.9395
Epoch 64: val_accuracy did not improve from 0.86167
225/225 [==============================] - 6s 28ms/step - loss: 0.1617 -
accuracy: 0.9393 - val_loss: 0.4438 - val_accuracy: 0.8533 - lr: 1.2500e-04
```

Training & Validation Loss/Accuracy Curves

[ ]:
```
import matplotlib.pyplot as plt
ep = [i for i in range(1,len(history.history['loss'])+1)]
curves = [history.history[i] for i in␣
 ↪['loss','val_loss','accuracy','val_accuracy']]
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(ep,curves[0],'o',ls='-',label='Training Loss',markevery=[-1])
plt.plot(ep,curves[1], label='Validation Loss')
l1 = (ep[-1],curves[0][-1])
plt.annotate(f"{l1[1]:.3f}",xy=l1)
l2 = (ep[-1],curves[1][-1])
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.grid()
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(ep,curves[2], label='Training Accuracy')
plt.plot(ep,curves[3], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.grid()
plt.legend()

plt.tight_layout()
plt.show()
# plots for new config model2
```



Saving Model

[ ]:
```
bmodel = keras.models.load_model("/content/drive/MyDrive/CNN/weights/
↪best_model_0.86167.h5")
```

Loading Test Dataset

[27]:
```
test_path='/content/drive/MyDrive/seg_test/seg_test/'
```

[30]:
```
classes = ['buildings','forest','glacier','mountain','sea','street']
```

[31]:
```
lentest={}
for n1 in classes:
    l=len(os.listdir(test_path+'/'+n1))
    lentest[n1]=l
lentest
```

[31]:
```
{'buildings': 437,
 'forest': 474,
 'glacier': 553,
 'mountain': 525,
 'sea': 510,
 'street': 501}
```

[32]:
```
testimg=0
for e1 in lentest:
    testimg+=lentest[e1]
testimg
```

[32]:
```
3000
```

[33]:
```
xtest=[]
ytest=[]
for label in classes:
    pth=os.path.join(test_path,label)
    for img in tqdm.tqdm(os.listdir(pth)):
          pic=cv2.imread(os.path.join(pth,img))
          xtest.append(pic)
          ytest.append(classes.index(label))
```

```
100%|      | 437/437 [00:03<00:00, 131.98it/s]
100%|      | 474/474 [00:04<00:00, 107.59it/s]
100%|      | 553/553 [00:04<00:00, 121.69it/s]
100%|      | 525/525 [00:04<00:00, 123.54it/s]
100%|      | 510/510 [00:04<00:00, 112.52it/s]
100%|      | 501/501 [00:04<00:00, 120.30it/s]
```

Pre-Processing on Test Dataset

[34]:
```
for i in range(len(xtest)):
    img = np.array(xtest[i])
    xtest[i] = cv2.resize(img,(128,128))
```

[35]:
```
xtest = np.array(xtest)/255
ytest = np.array(ytest)
np.save('testX',xtest)
np.save('testY',ytest)
```

[36]:
```
import numpy as np
xtest = np.load("/content/testX.npy")
ytest = np.load("/content/testY.npy")
```

[37]:
```
ytest
```

[37]:
```
array([0, 0, 0, …, 5, 5, 5])
```

Results of Model Testing

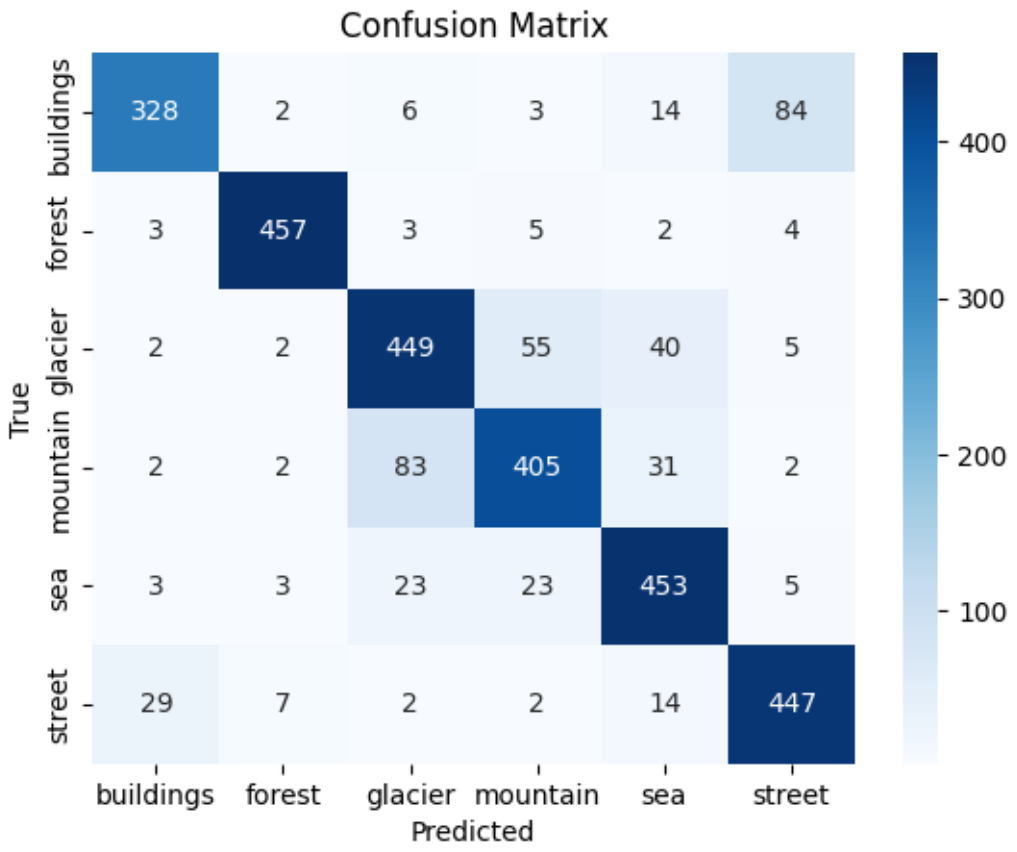
[38]:
```
ypred = bmodel.predict(xtest,batch_size=32)
```

```
94/94 [==============================] - 1s 7ms/step
```

[39]:
```
ypredi = np.argmax(ypred,axis=1)
ypredi
```

[39]:
```
array([0, 0, 0, …, 5, 5, 5])
```

[40]:
```
from sklearn.metrics import␣
 ↪confusion_matrix,classification_report,accuracy_score,f1_score
cm = confusion_matrix(ytest,ypredi)
sns.heatmap(cm, annot=True, fmt="d", cmap='Blues', xticklabels=classes,␣
 ↪yticklabels=classes)
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")
plt.show()
```



[41]:
```
acc = accuracy_score(ytest,ypredi)
print(f"Accuracy = {acc*100:.2f}%")
```

```
Accuracy = 84.63%
```

[42]:
```
f1 = f1_score(ytest,ypredi,average='macro')
print(f"F1-Score = {f1*100:.2f}%")
```

```
F1-Score = 84.73%
```

[43]:
```
rep = classification_report(ytest,ypredi,target_names=classes)
print(rep)
```

```
              precision    recall  f1-score   support

   buildings       0.89      0.75      0.82       437
      forest       0.97      0.96      0.97       474
     glacier       0.79      0.81      0.80       553
    mountain       0.82      0.77      0.80       525
         sea       0.82      0.89      0.85       510
```

```
          street       0.82      0.89      0.85       501

        accuracy                           0.85      3000
       macro avg       0.85      0.85      0.85      3000
    weighted avg       0.85      0.85      0.85      3000
```

[ ]: